Retargetting Example Sounds to Interactive Physics-Driven Animations

Cécile Picard, Nicolas Tsingos, François Faure

Retargetting Example Sounds to Interactive Physics-Driven Animations

Cécile Picard[1], Nicolas Tsingos[1,2], and François Faure[3]

[1]*REVES/INRIA Sophia-Antipolis, BP 93 F-06902 Sophia Antipolis, France*

[2]*now with Dolby Laboratories, San Francisco, CA, USA*

[3]*INRIA Rhône-Alpes, Laboratoire Jean Kuntzmann, Université de Grenoble and CNRS, 38330 Montbonnot Saint Martin, France*

Correspondence should be addressed to Cécile Picard (`cecile.picard@sophia.inria.fr`)

ABSTRACT

This paper proposes a new method to generate audio in the context of interactive animations driven by a physics engine. Our approach aims at bridging the gap between direct playback of audio recordings and physically-based synthesis by retargetting audio grains extracted from the recordings according to the output of a physics engine. In an off-line analysis task, we automatically segment audio recordings into atomic grains. The segmentation depends on the type of contact event and we distinguished between impulsive events, e.g. impacts or breaking sounds, and continuous events, e.g. rolling or sliding sounds. We segment recordings of continuous events into sinusoidal and transient components, which we encode separately. A technique similar to matching pursuit is used to represent each original recording as a compact series of audio grains. During interactive animations, the grains are triggered individually or in sequence according to parameters reported from the physics engine and/or user-defined procedures. A first application is simply to reduce the size of the original audio assets. Above all, our technique allows to synthesize non-repetitive sounding events and provides extended authoring capabilities.

1. INTRODUCTION

Audio rendering of complex contact interactions between objects animated with physics engines is becoming a key challenge for interactive simulation and gaming applications. Contact sounds should not appear repetitive, as it is often the case when a small number of pre-recorded samples is directly used. Increasing the number of samples is not always possible due to memory constraints. Furthermore, matching sampled sounds to interactive animation is difficult and often leads to discrepancies between the simulated visuals and their soundtrack. Alternatively, contact sounds can be automatically generated using sound synthesis approaches. However, interactive physically-based synthesis currently targets limited classes of sounds and, to date, cannot render convincing breaking, tearing or non-rigid interaction sounds.

Our approach aims at bridging the gap between direct playback of sound samples and physically-based synthesis by automatically analyzing audio recordings so that they can be retargetted to interactive animations. It combines an off-line analysis process with an interactive online resynthesis as illustrated in Figure 1. In the off-line process, we automatically segment audio recordings into atomic time-slices or *grains* and build a shared, compact dictionary. Based on a correlation estimation, we represent each original recording as a series of grains. During interactive animations, the audio grains are triggered individually or in sequence according to parameters reported from the physics engine and/or user-defined procedures.

Our specific contributions are:

- a method for automatic analysis of audio recordings, and the generation of compact dictionaries of audio grains and correlation patterns.
- a specific technique for analysis of recordings of continuous contact events such as rolling or sliding, that separately encodes the steady-state and transient parts.

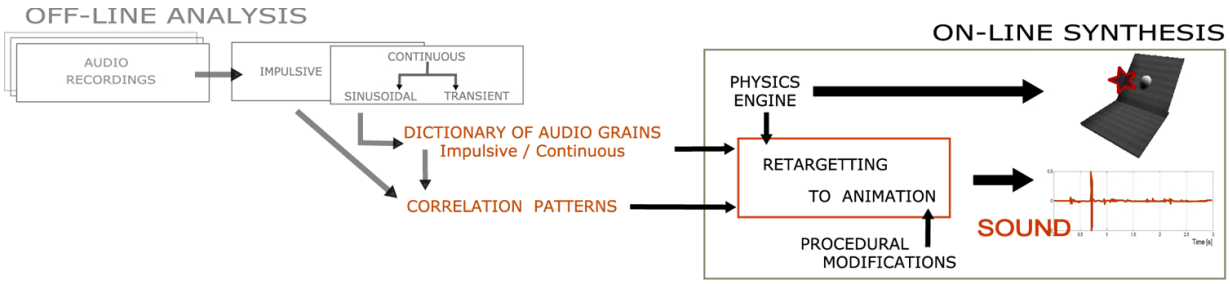

Fig. 1: *Overview of our approach combining off-line analysis of recorded sounds with interactive retargetting to motion.*

- a solution for generating on-line audio for interactive animations by retargetting the audio grains in order to match inter-object contacts and interaction state.

The proposed framework allows the generation of compelling and controllable soundtracks in the context of physics-driven animations. Key aspects of our approach are its low memory usage and the different levels of authoring available to both audio programmers and designers.
The remainder of this paper is organized as follows. We first review related work in Section 2. In Section 3, we detail our off-line analysis including the segmentation of source recordings for extraction of audio grains and the generation of correlation patterns. Section 4 details the retargetting procedure and discusses several strategies. In Section 5, we present some results obtained with our method. Finally, in Section 6 we discuss limitations of our approach and propose possible extensions before concluding.

2. PREVIOUS WORK

Interactive Generation of Audio: The traditional approach to creating soundtracks for interactive physically-based animations is to directly play-back pre-recorded samples e.g., synchronized with the contacts reported from a rigid-body simulation. Due to memory constraints, the number of samples is limited, leading to repetitive audio. Moreover, matching sampled sounds to interactive animation is difficult and often leads to discrepancies between the simulated visuals and their companion soundtrack. Finally, this method requires each specific contact interaction to be associated with a corresponding pre-recorded sound, resulting in a time-consuming authoring process.

Physically-based Sound Synthesis: Previous work in computer graphics [15, 18, 24, 25] and computer music [7] has explored methods for generating sound based on physical simulation. Most approaches target sounds emitted by vibrating solids. For interactive simulations, a widely used solution is to synthesize vibration modes whose parameters, i.e frequencies, corresponding gains and decay rates, are obtained through modal analysis. Modal data can be obtained from simulations [15, 18] or extracted from recorded sounds of real objects [25]. In order to speed-up modal synthesis, a fast approach that exploits the inherent sparsity of modes in the frequency domain has recently been introduced in [5]. However, these methods require significant parameter tuning to achieve realistic results, making them challenging to use in a game production pipeline. In addition, they currently target limited classes of sounds and, to date, cannot render convincing breaking, tearing or non-rigid interaction sounds.

Sound Texture Modeling: Sound texture modeling allows for a more flexible playback of sound samples and is a widely used concept in computer music.
Among the large number of sound texture generation methods, concatenative synthesis [19, 20] is closest in spirit to our work. Concatenative synthesis approaches aim at generating a meaningful macroscopic waveform structure from a large number of shorter waveforms. They typically use databases of sound snippets, or grains, to assemble a given target phrase. In signal processing for music, [12] presents a decomposition of music signals into a small number of atoms that retain some of the most salient features of the audio data. In [13], one of the first graphical interfaces for real-time granular synthesis is proposed allowing the user to model the sound grains and their distribution. Miller Puckette [17] proposes a technique to drive the synthesis based on a real-

time audio-stream used as a source of timbral control. Finally, [9] introduces a method to render aerodynamic sounds according to the motion of objects or wind velocity. Sound textures based on computational fluid dynamics were used.

Authoring and Interactive Control: Cook [6] introduces an automatic analysis and parametric synthesis of walking and other (gesture-rate) periodically modulated noisy sounds that is related to our approach. The method consists in evaluating recordings of walking by extracting characteristic features as tempo, basic resonances and control envelopes. However, the approach remains limited in the classes of sounds it can handle.

3. ANALYSIS OF EXAMPLE SOUNDS

In this section, we detail the components of our automatic analysis, performed off-line. The analysis process consists in segmenting the available audio recordings into audio grains and generating corresponding resynthesis information.

3.1. Impulsive and Continuous Contacts

We distinguished between impulsive and continuous contacts. Audio recordings of impulsive contacts, such as impact, hitting or breaking sounds, are characterized by the preponderance of transients. Conversely, recordings of continuous contacts, such as rolling or sliding, contain a significant steady-state part. This steady-state part is characteristic of the objects being continuously excited during contact and should be preserved during resynthesis. Continuous contact sounds are thus segmented into a steady-state and a noise/transient part. Segmentation is carried out using the Spectral Modeling Synthesis (SMS) approach of [21]. SMS is based on modeling sounds as stable sinusoids (sinusoidal part) plus noise (residual part). The analysis procedure extracts the steady-state part by tracking the time-varying spectral characteristics of the sound and represents them with time-varying sinusoids. These sinusoids are then subtracted from the original sound. The residual part mainly contains energy at high-frequencies and may represent the micro-transient character of the event. Thus, we will refer to the residual part as the *transient part*. As a result, recordings of continuous-contacts are first decomposed into two recordings, prior to grain extraction while recordings of impulsive contact are directly segmented.

3.2. Automatic Extraction of Audio Grains

In an off-line analysis, audio recordings are fragmented into syllable-like audio grains. Audio grains are elementary signal components (typically 300 to 3000 samples long, i.e. 0.01 to 0.1 s @ 44.1KHz) and, for instance, may correspond to discrete impacts in the more complex recording of a breaking glass.

Audio recordings are by nature non-stationary. Thus, their amplitude and frequency are time-varying and the variations are characteristic of the signal. In signal processing, the *spectral flux* is a measure of how fast the power spectrum of a signal is changing and can be calculated as the Euclidean distance between the two normalized spectra of consecutive frames. The spectral flux is typically used for onset detection [8], and we conjecture that its variations are appropriate to extract the audio grains. In our implementation, the spectral flux is calculated as in [10] and is applied to all the input recordings. For recordings of continuous contacts, we calculate spectral flux on both sinusoidal and transient parts. The recordings are then segmented into grains at the inflection points of the spectral flux, as shown in Figure 2.

Extracted audio grains are windowed with a Tukey window, i.e. a cosine-tapered window, with r=0.02 for the ratio of taper and normalized for power conservation.

To limit the number of extracted grains, we discard the grains with low energy according to a user-defined threshold. The remaining audio grains are labelled either continuous or transient depending on their source recording and stored away.

As a result, we obtain a dictionary of grains that can be retargetted to parameters of the physics engine and/or user-defined procedures. Note that the dictionary is non-redundant since the segmentation occurs exactly at the inflection points of the spectral flux. Audio grain extraction can be carried out on an entire database of audio recordings, leading to a large dictionary that offers more possibilities than the original recordings alone.

3.3. Generation of Correlation Patterns

In the second step of the analysis process, original audio recordings are decomposed onto the "basis" formed by the previously extracted audio grains to obtain correlation patterns. The correlation patterns will be used to encode each original recording as a series of audio grains.

Our method is similar in spirit to matching pursuit (MP) approaches [14] that decompose any signal into a sparse

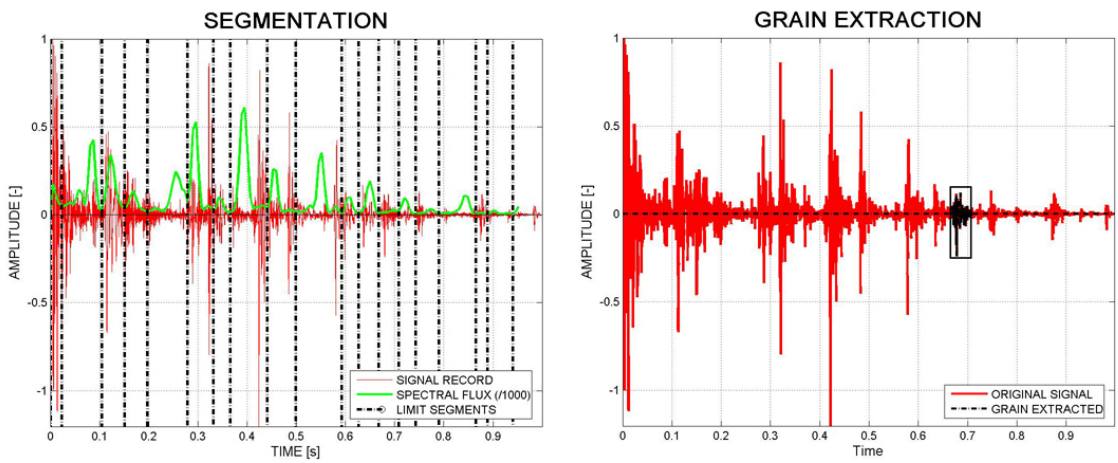

Fig. 2: *Segmentation of an impulse-like sound event according to a spectral flux estimation (left) and example of an extracted audio grain (right).*

linear expansion of waveforms selected from a redundant dictionary of functions [11] by performing iterative local optimization. MP approaches are traditionally considered too slow for high-dimensional signals, with impractical runtimes. In comparison with traditional MP approaches, we introduce a greedy grain selection approach avoiding the iterative process.

The cross-correlation between each available recording and each grain is calculated. Normalized vectors are used for the calculation. We greedily identify major peaks of the cross-correlation function using a peak picking step. A user-defined number of peaks is then preserved for each recording (for instance the k largest). A correlation pattern is stored as the list of retained cross-correlation peaks across all grains, with the corresponding time indices: they represent the appropriatness of each grain to synthesize the considered original source. Cross-correlation calculation is performed for all recordings, including the sinusoidal and transient part of continous contact sounds.

A direct benefit of our analysis step is a compact representation of the original sound database since the dictionary of grains and the correlation patterns typically have a smaller memory footprint than the original assets.

4. FLEXIBLE SOUND SYNTHESIS

Once the analysis has been performed, we can resynthesize an infinite variety of sounds similar but not identical to the source recordings. We introduce a flexible *audio shading* approach [23] allowing us to render contact sounds for animation based on collision reporting from the real-time physics engine and/or user-defined procedures.

4.1. Resynthesis of the Original Recordings

The correlation patterns obtained in the off-line analysis process can be directly used to reconstruct the original recordings at run-time.

The resynthesis is performed by choosing from the correlation pattern the grains presenting the maximum correlation amplitude (see Figure 3). Similar to concatenative synthesis, our approach considers candidate grains one at a time, in a time-interval starting from the end of the previously added grain. In order to take into account the effect of windowing (see Section 3.2), the look-up starts slightly before the end of the previously added grain. In our case, this offset is consisent with the overlap of the Tukey window and equals (r**len*/2) where *len* is the length of the previously added grain and r is the taper of the window (here, r=0.02). The candidate grain is searched within an interval of duration equal to half of the median length of the available grains. Recall from section 3.3 that several correlation peaks might have been stored within that time interval. To reconstruct a signal closest to the original recording, the grain with maximum correlation value is chosen. We then concatenate the grain using overlap-add blending at the specific correlation time index, with an amplitude equal to the power of the original audio recording in the considered

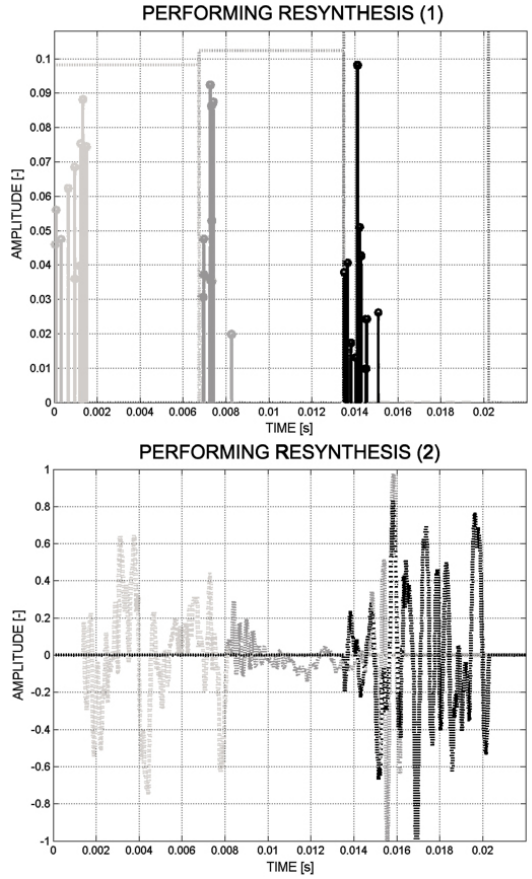

Fig. 3: *Concatenating grains for resynthesis. Top: example correlation peaks for the considered recording. Bottom: Reconstructed recording. In this case, only the grains presenting the maximum correlation amplitude with the considered original recording are chosen.*

time-window normalized by the power of the grain.

4.2. Physics Parameters for Retargetting

For a compelling re-synthesis, it is necessary to match the audio content to the contact parameters reported by the physics engine. In particular, we need to detect and distinguish impulsive and continous contact, including rolling and sliding. While existing physics engines, such as *Havok* or *Nvidia's PhysX*, do report contacts, they generally fail to distinguish between rolling and sliding, making it challenging to trigger consistent sound effects.

Contacts between bodies have been extensively investigated for sound rendering [4, 16]. Sliding involves multiple micro-collisions at the contact area. For rolling, the surfaces have no relative speed at the contact point and this difference in contact interaction leads to an audible change. Based on previous studies on contact simulations, we analyze penetration forces and relative velocities across time in order to detect impulsive and continuous contacts, as seen in Figure 4.

When a contact is reported, grains are appropriately retargetted according to the labelling of the grain obtained during the analysis (see Section 3). For impacts, impulsive grains are retargetted to penetration force peaks, knowing that relative velocity peaks occur at the same time instant as penetration force peaks.

Continuous contacts are detected when the penetration force and the relative velocity are constant. Rolling is identified when the relative velocity is equal or close to zero. Audio grains can be chosen either randomly or retargetted from a consistent correlation pattern, for instance a pattern extracted from a rolling sound if rolling was detected. They are concatenated with an amplitude proportional to the power of the contact interaction, i.e. the dot product of the penetration force and the relative velocity. For continuous contacts, spectral domain modifications can also be easily achieved, thanks to the SMS approach (see Section 3.1). For instance, the sound can be adapted to the velocity of the objects in interaction. In addition, the SMS approach enables the modification of the continuous and transient parts separately. In [22] the effect of surface roughness on the frequency of non-squealing frictional sound generated in dry flat/flat sliding contact was studied. It was found that rubbing frequency or load does not qualitatively change the spectrum of the sound but high rubbing speed generates higher levels of power spectral density and high load gives much broader peaks. Thus, transient grains are played back without any modification in frequency content. In contrast, continous grains are frequency-scaled according to the velocity of the interaction, shifting towards higher frequencies as the velocity increases. Examples of modified recordings of continuous contact events are provided as additional material [1].

4.3. Flexible Audio Shading Approach

Our approach also supports additional, user-defined, resynthesis schemes. For instance, time-scaling of the original audio recordings can be easily implemented using the previously calculated correlation patterns and an approach similar to Section 4.1. First, a time-scale modification factor α is set: $\alpha < 1$ to reduce the length of the original recording and $\alpha > 1$ to increase it. The length

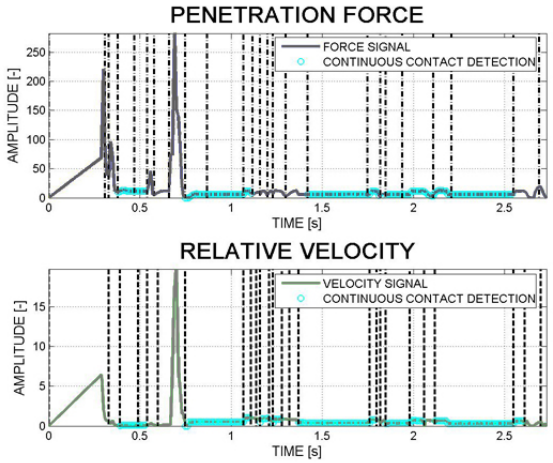

Fig. 4: *Detection of continuous contact based on the evaluation of the penetration force and the relative velocity reported by the physics engine.*

of the time-scaled recording is α times the length of the original audio recording. The correlation patterns that code the original recordings as series of audio grains (see Section 4.1) are used to construct the modified recording. Series of grains are displaced in time by a factor of α. Note that, in the case of time-stretching, this approach does not only shift original grains in time but also synthesizes additional material to fill in the gaps introduced by the stretching process.

The method for concatenating grains is flexible and allows the synthesis of an infinity similar audio events. Similar audio events can be built using a variety of approaches. For instance, variable audio content can be matched to the characteristic rhythmic pattern of a source recording using its correlation pattern. Time-scaled audio recordings and synthesis of similar audio events are provided for listening as additional material [1].

5. RESULTS

We tested our approach using a complete database of recorded contact interaction events used in a commercial video game. The database comprises various material types, e.g., plastic, metal, glass, and interaction types, such as hitting, breaking or rolling. It consists in 855 audio files (44.1 KHz, 16 bit PCM data), for a total recording time of 18 minutes ($\approx$ 48 million samples) and a memory footprint of 95 Mb. We assumed that those recordings are free of outside noise and unwanted echoes in order to extract audio grains that can be retargetted to a wide range of animations. The analysis of the recordings required to extract the grains and generate the correlation patterns was performed in an off-line process using a *MATLAB* implementation (see Section 3).

5.1. Dictionary-based compression

A first application of our approach is simply to reduce the memory footprint of the original assets. We applied our approach to a single group of recordings, the stone interaction events with 94 recordings in total, representing 14,6Mb of data. We extracted 5171 audio grains from which 3646 were impulsive and 1525 were continuous, and computed the correlation patterns for all input recordings. The extraction of the grains for the 94 recordings took 11 minutes while the correlation patterns were computed in 4 hours.

In this case, we only coded into the correlation patterns the grains with maximum correlation amplitude. A single grain was encoded for each time-interval multiple of the average grain length. As an example, if we consider an original recording of 3 sec. and grains with an average length of 3000 samples (0.1 s @ 44.1KHz), the correlation pattern consists in about 45 peaks, taking into account the overlap of the grains. The memory footprint was 11,64Mb for the audio grains and 29Kb for the correlation patterns, leading to a 20% gain over the original assets in this case. The gain in memory footprint is not high but still significant and the approach allows us to create more content than the original database. In addition, to more efficiently manage the memory usage, only a subset of the grain dictionary can be made available, such as the grains showing the highest correlation with the type of recordings we consider. A subsequent lossy audio compression step on the obtained dictionaries of grains would lead to an additional 5:1 (e.g., 128 kbps mp3) to 10:1 (e.g., mp3 VBR) compression ratio.

We provide as additional material a number of comparisons between original audio recordings and reconstructed versions in order to evaluate the quality of the approach [1].

5.2. Interactive physics-driven animations

Retargetting of extracted audio grains was used to generate interactive audio in the context of simulations driven by the *Sofa* [3] physics engine. We used the precomputed correlation patterns generated in the analysis step. The penetration force and the relative velocity of the objects

in interaction were studied across time in order to detect impacts and continuous contacts. The grains were chosen from an appropriate correlation pattern and triggered, in a random manner or in series following the approach described in Section 4.2.

Physics engines typically report contact information at a framerate of 30 to 60Hz, whereas sound has to be synthesized at 44kHz. In the case where the grains are of minimum length, i.e. 300 samples, the physics framerate is not sufficient to guarantee a continuous sound rendering. A benefit of our approach is that we can continuously select and play-back grains from a given correlation pattern to ensure a continous result, regardless of the frame-rate of the physics simulation.

Example movie files demonstrating our approach are provided as additional material [1]. The result shows that our approach can be effectively used for plausible soundtracks consistent with visual rendering, although more work is required to perfectly interpret the parameters delivered by the physics engine.

6. CONCLUSION

We have proposed a technique to generate in real-time plausible soundtracks, synchronized with the animation, in the context of simulations driven by a physics engine. Our approach dynamically retargets audio grains to the events reported by the engine. We proposed an off-line solution to automatically extract grains and correlation patterns from a database of recordings.

Our method can be applied to reduce the footprint of the original assets and can further be combined with traditional compression techniques while maintaining acceptable audio quality. In addition, the approach allows trading memory footprint for variety at synthesis time and frees the sound designer from the hassle of having to create a variety of similar sounds by hand. By combining audio content and information that describes how that content is to be used, our approach might be a good candidate for the Interactive XMF Audio File Format (eXtensible Music Format) [2].

The main limitation of our technique is the preprocessing time required to calculate the correlation between the original recordings and the grains. However, this could be improved by implementing this step in C++.

We believe our work addresses important issues for both audio programmers and designers by offering extended sound synthesis capabilities in the framework of standard sample-based audio generation.

In the future, higher level statistical analysis of the patterns obtained for similar recordings could lead to more efficient and controllable resynthesis. Finally, clustering of similar grains could be investigated to further reduce the size of the data structures and the pre-processing time.

7. ACKNOWLEDGEMENTS

This work was partially funded by *Eden Games*, an *ATARI* game studio. We would like to thank David Alloza and *Eden Games* for their feedback at various stages of this project.

8. REFERENCES

[1] Additional Material.
`http://evasion.imag.fr/~Cecile.Picard/SupplementalAES`.

[2] Interactive XMF Audio File Format (eXtensible Music Format).
`http://www.iasig.org/wg/ixwg/`.

[3] Simulation Open Framework Architecture.
`http://www.sofa-framework.org/`.

[4] F. Avanzini, M. Rath, and D. Rocchesso. Physically-based audio rendering of contact. In *ICME '02: Proceedings of the IEEE International Conference on Multimedia and Expo*, volume 2, pages 445–448, 2002.

[5] Nicolas Bonneel, George Drettakis, Nicolas Tsingos, Isabelle Viaud-Delmon, and Doug James. Fast modal sounds with scalable frequency-domain synthesis. In *SIGGRAPH '08: ACM SIGGRAPH 2008 papers*, pages 1–9, New York, NY, USA, 2008. ACM.

[6] Perry R. Cook. Modeling bill's gait: Analysis and parametric synthesis of walking sounds. In *AES 22nd Intl. Conf. Virtual, Synthetic and Entertainment Audio*, pages 73–78, Espoo, Finland, 2002.

[7] Perry R. Cook. *Real Sound Synthesis for Interactive Applications*. A. K. Peters, 2002.

[8] S. Dixon. Onset detection revisited. In *Proceedings of the 9th International Conference on Digital Audio Effects (DAFx-09)*, Montreal, Canada, sept 2006.

[9] Yoshinori Dobashi, Tsuyoshi Yamamoto, and Tomoyuki Nishita. Real-time rendering of aerodynamic sound using sound textures based on computational fluid dynamics. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 732–740, New York, NY, USA, 2003. ACM.

[10] Dubnov. Catbox: Computer audition toolbox in Matlab v0.1, 2006. online web resource.

[11] Garry Kling and Curtis Roads. Audio Analysis, Visualization, and Transformation with the Matching Pursuit Algorithm . In *Proceedings of the 7th International Conference on Digital Audio Effects (DAFx-04)*, pages 33–37, Naples, Italy, 2004.

[12] P. Leveau, E. Vincent, G. Richard, and L. Daudet. Instrument-specific harmonic atoms for mid-level music representation. *Audio, Speech, and Language Processing, IEEE Transactions on*, 16(1):116–128, 2008.

[13] Daniel Lopez, Francesc Marti, and Eduard Resina. Vocem: An application for real-time granular synthesis. In *Digital Audio Effects (DAFx) Conference*, 1998.

[14] S. Mallat and Z. Zhang. Matching pursuits with time-frequency dictionaries. *IEEE Transactions on Signal Processing*, 41(12):3397–3415, 1993.

[15] James F. O'Brien, Chen Shen, and Christine M. Gatchalian. Synthesizing sounds from rigid-body simulations. In *SIGGRAPH '02: ACM SIGGRAPH Symposium on Computer Animation*, pages 175–181, San Antonio, Texas, 2002. ACM Press.

[16] Dinesh Pai, Kees van den Doel, Doug James, Jochen Lang, John E. Lloyd, Joshua L. Richmond, and Som H. Yau. Scanning physical interaction behavior of 3d objects. In *Proc. SIGGRAPH'01*, pages 87 – 96, August 2001.

[17] M. Puckette. Low-dimensional parameter mapping using spectral envelopes. In *ICMC '04: Proceedings of International Compupter Music Conference*, Miami, USA, 2004. International Computer Music Association.

[18] Nikunj Raghuvanshi and Ming C. Lin. Interactive sound synthesis for large scale environments. In *SI3D'06: Proceedings of the 2006 symposium on Interactive 3D Graphics and Games*, pages 101–108. ACM Press, 2006.

[19] Curtis Roads. Asynchronous granular synthesis. *Representations of musical signals*, pages 143–186, 1991.

[20] Diemo Schwarz. Concatenative sound synthesis: The early years. *Journal of New Music Research*, 35(1):3–22, 2006.

[21] X. Serra. *Musical Sound Modeling with Sinusoids plus Noise*, pages 91–122. Swets and Zeitlinger, 1997.

[22] Boyko L. Stoimenov, Suguru Maruyama, Koshi Adachi, and Koji Kato. The roughness effect on the frequency of frictional sound. *Tribology International*, 40(4):659 – 664, 2007. NORDTRIB 2004.

[23] Tapio Takala and James Hahn. Sound rendering. *ACM Computer Graphics, SIGGRAPH'92 Proceedings*, 28(2), July 1992.

[24] Kees van den Doel, Paul G. Kry, and Dinesh. K. Pai. Synthesis of shape dependent sounds with physical modeling. In *Proceedings of the International Conference on Auditory Display (ICAD)*, 1996.

[25] Kees van den Doel, Paul G. Kry, and Dinesh. K. Pai. Foley automatic: physically-based sound effects for interactive simulation and animation. In *Proc. SIGGRAPH '01*, pages 537–544, New York, NY, USA, 2001. ACM Press.